Face Cartonization

- **Used algorithms:**
    - Viola Jones Algorithm to detect face location
    - (Average of Synthetic Exact Filters) ASEF Algorithm to detect facial features.
        - We chose this algorithm because it archives high accuracy and real time applicable which is perfect for our scenario. Other algorithms like Gabor, Cascade or Mean are more computing intensive without significant improvement.
    - K-Means
        - Color compression for cartonization

- **Experiment results and analysis:**
    - Viola Jones
        - Training data set consists of 3000 faces and 300 non faces.
        - Using 10 weak classifiers for Adaboostsing.
        - Using 40 strong classifiers for Viola Jones.
        - Metrics: [test]
            - True_positive: 6,686
            - True_negative: 4,227
            - False_positive: 38
            - False_negative: 375
            - Accuracy: 0.964
            - Precision: 0.994
            - Recall: 0.947
            - Specify: 0.991
    - ASEF
        - Training 1.5k images takes around one minute.
        - Testing on 500 unseen data takes around 20 seconds with:
            - Left Eye accuracy: 95.6%
                - correct: 478, false: 22
            - Right Eye accuracy: 96.2%
                - correct: 481, false: 19
            - Nose accuracy: 92.2%
                - correct: 461, false: 39
            - Mouth accuracy: 94.4%

- correct: 472, false: 28

- Testing on 500 unseen data takes around 20 seconds with:
  - Left Eye accuracy: 92.46119733924613%
    - correct: 417, false: 34
  - Right Eye accuracy: 95.34368070953437%
    - correct: 430, false: 21
  - Nose accuracy: 91.7960088691796%
    - correct: 414, false: 37
  - Mouth accuracy: 92.46119733924613%
    - correct: 417, false: 34
- Algorithm accuracy compared to dlib 68 face feature extractor with around 10% location change.

- **Conclusion and references:**
  - ➢ viola jones explanation
  - ➢ viola jones paper
  - ➢ Bolme, D. S., Draper, B. A., & Beveridge, J. R. (2009, June). Average of synthetic exact filters. In 2009 IEEE Conference on Computer Vision and Pattern Recognition (pp. 2105-2112). IEEE.

- **Work division:**

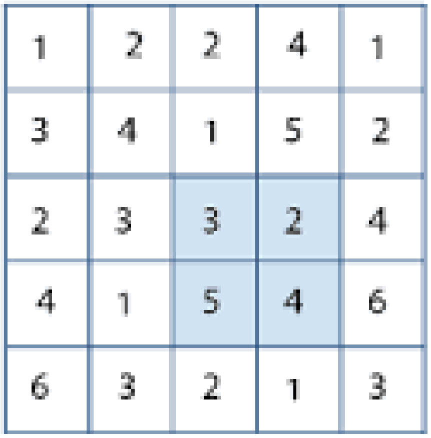
Name	Section	BN	Work load
Karim Mahmoud Kamal	**2**	**10**	**Calculate Integral image, Extract Haar features, Determine and apply Haar-like features**
Mohamed Abdulhady	**2**	**16**	**ASEF implementation, K-Means, Cartoon filter cropping, testing, training, Data Collection, non-cartoon filter, data cleaning**
Mohamed Kotb	**2**	**17**	**Weak and strong classifiers, Adaboost, training and multiscale classification**
Mohamed Kamal	**2**	**18**	**ASEF implementation, K-Means, testing, training, Data Collection, non-cartoon filter, GUI**

- **Additional comments:**

Face Detection Using Viola-Jones Algorithm

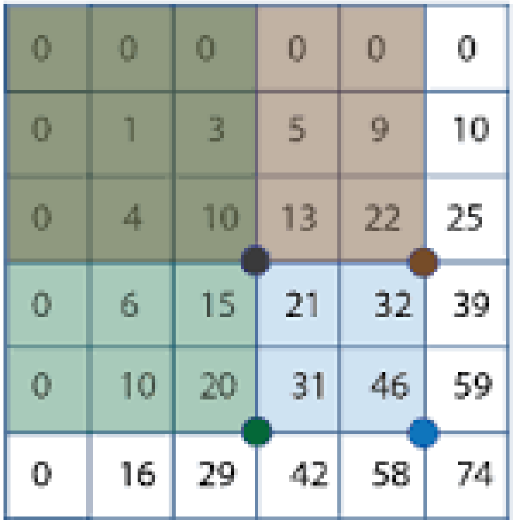
1- Calculating Integral Image:

To simplify the calculations of haar-like features and save the time of iterating over all pixels.



1	2	2	4	1
3	4	1	5	2
2	3	3	2	4
4	1	5	4	6
6	3	2	1	3

input image



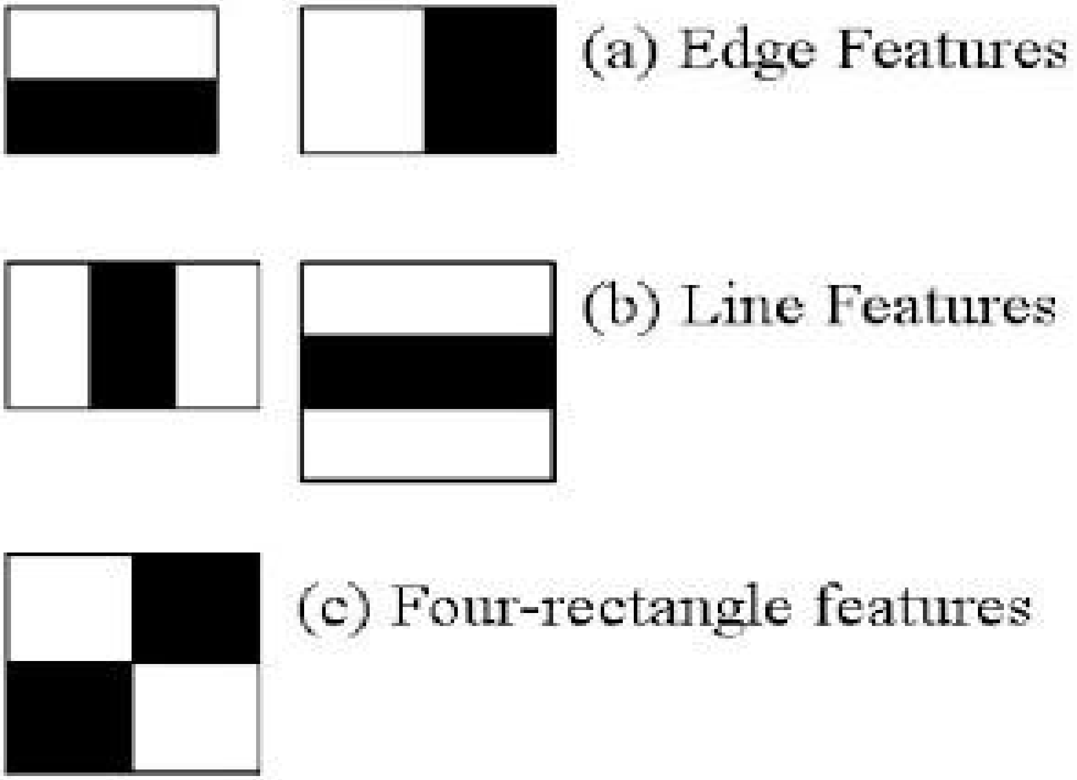
0	0	0	0	0	0
0	1	3	5	9	10
0	4	10	13	22	25
0	6	15	21	32	39
0	10	20	31	46	59
0	16	29	42	58	74

integral image

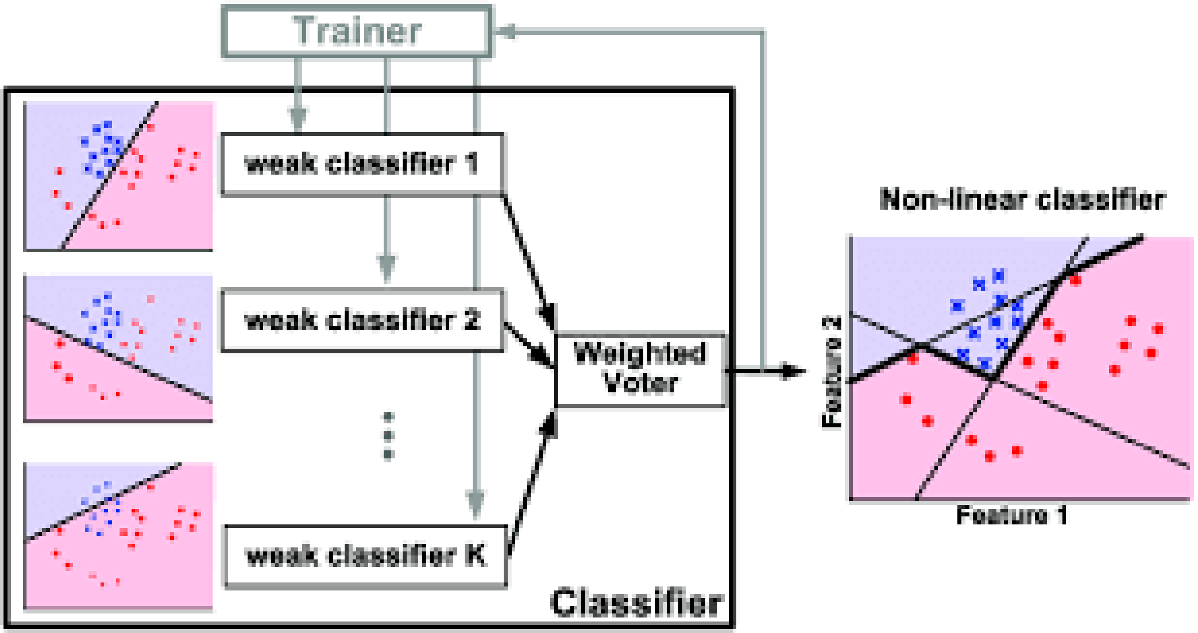
2- Calculating Haar-like features:

The haar feature windows we used in our implementation are:

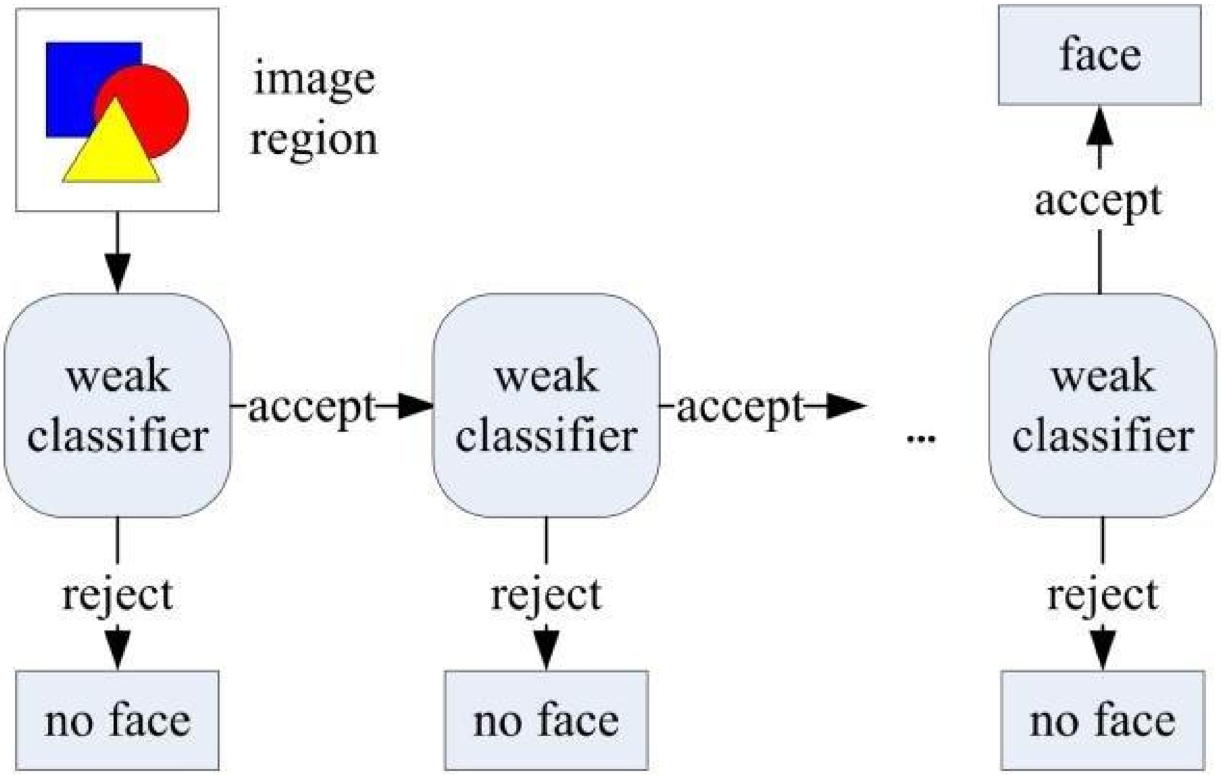
- **2 horizontal**
- **2 vertical**
- **3 horizontal**
- **3 vertical**
- **2*2 diagonal**



3- AdaBoost Learning Algorithm:



4- Cascade Filter:



Dataset:

The data is described at http://cbcl.mit.edu/software-datasets/FaceData2.html, and downloaded from www.ai.mit.edu/courses/6.899/lectures/faces.tar.gz and compiled into pickle files.

ASEF

$$g_i(x,y) = e^{-\frac{(x-x_i)^2+(y-y_i)^2}{\sigma^2}}$$

$$g(x,y) = (f \otimes h)(x,y) = \mathcal{F}^{-1}(F(\omega,\nu)H(\omega,\nu))$$

Calculate convolution filter

$$H_i^*(\omega,\nu) \quad = \quad \frac{G_i(\omega,\nu)}{F_i(\omega,\nu)}$$

Average all filters

$$H_\mu^*(\omega,\nu) \quad = \quad \frac{1}{N}\sum_{i=1}^{N} H_i^*(\omega,\nu)$$

$$h_\mu(x,y) \quad = \quad \frac{1}{N}\sum_{i=1}^{N} h_i(x,y)$$

Get feature location from image

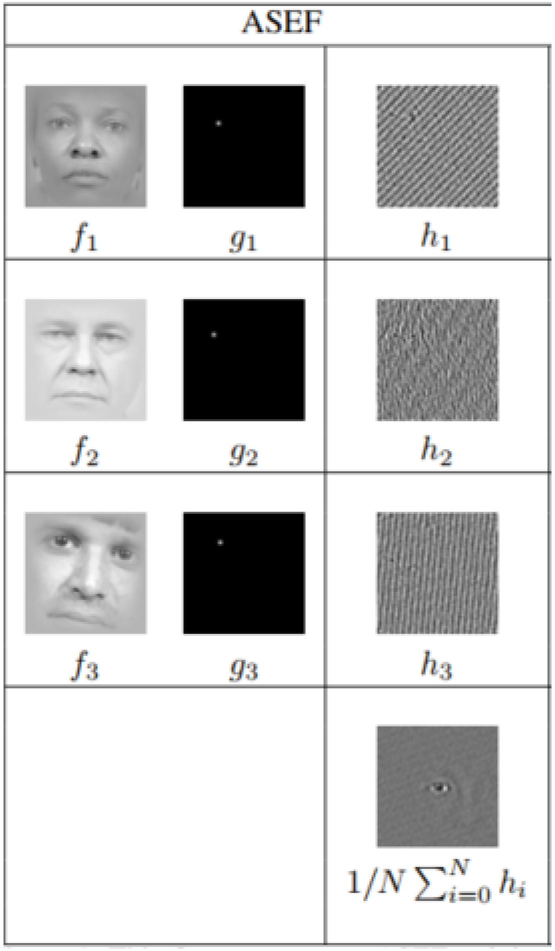
$$G(\omega,\nu) \quad = \quad F(\omega,\nu)H^*(\omega,\nu)$$

Dataset:

Celebrity and Cascaded faces datasets from kaggle. Each image was annotated with feature coordinates using dlib. Spurious images were cleaned.

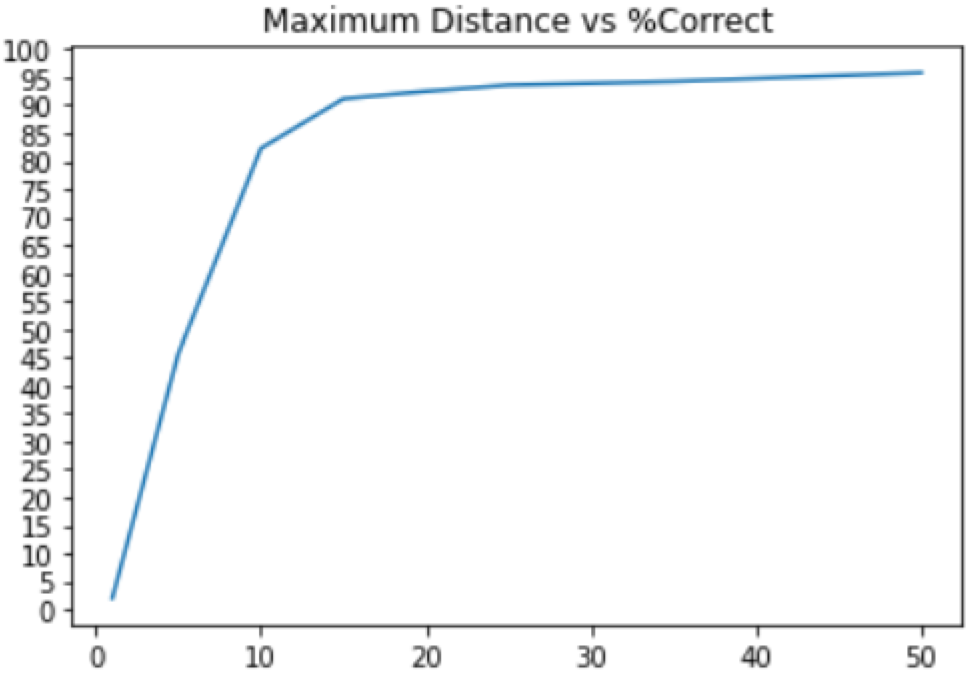
- **https://mmlab.ie.cuhk.edu.hk/projects/CelebA.html**
- **https://www.kaggle.com/datasets/stoicstatic/face-recognition-dataset?resource=download-directory**

The next figures illustrates the gaussian filter responding to a left eye in an image:



That’s why the algorithm is able to achieve high performance because it relies on multiplication and convolution properties so the final trained filter is a gaussian filter which can be multiplied in frequency domain (convolution in spatial domain) to then achieve our maximum interest point which is the feature location.

The effect of distance change on accuracy



Weakness points:

- Extreme rotations will cause face detection and facial feature detection to fail.
- Facial feature algorithms will have a hard time detecting eye locations while wearing dark glasses note the algorithm has no problem detecting eyes while closed.